Reading Challenging Barcodes with Cameras

Orazio Gallo and Roberto Manduchi

{orazio, manduchi}@soe.ucsc.edu, University of California, Santa Cruz.

Abstract

Reading barcodes with standard cameras, such as cell phone cameras, enables interesting opportunities for ubiquitous computing. Unfortunately, current camera-based barcode readers do not work well when the image has low resolution, is out of focus, or is blurred due to motion. One main reason for this poor performance is that virtually all existing algorithms perform some sort of binarization, either by gray scale thresholding or by finding the bar edges. We propose a new approach to barcode reading that never needs to binarize the image. Instead, we use deformable barcode digit models in a maximum likelihood setting. We show that the particular nature of these models enables efficient integration over the space of deformations. Global optimization over all digits is then performed using dynamic programming. Experiments with challenging UPC-A barcode images show substantial improvement over other state-of-the-art algorithms.

1. Introduction

Virtually every packaged good is labeled with at least one form of barcode, generally a flavor of either the EAN or the UPC standards. The success of barcode technology for identification, tracking, and inventory derives from its ability to encode information in a compact fashion with very low associated cost.

Barcode reading via dedicated scanners is a mature technology. Commercial laser-based, hand-held barcode scanners achieve robust reading with a reasonable price tag. Recently, however, there has been growing interest in accessing barcodes with regular cellphones rather than with dedicated devices. Since cellphones are of ubiquitous use, this would enable a multitude of mobile applications. For example, a number of cellphone apps have appeared recently that provide access to the full characteristics and user reviews for a product found at a store, or to on-line price comparisons via barcode reading.

Unfortunately, images taken by cellphone cameras are often of low quality. Many cellphone cameras on the market are equipped with low-grade lenses, generally lacking focusing capability, which frequently produce blurred images. Only few cellphones have flashes so motion blur and noise are an extremely common problem for pictures taken in low light conditions. All of these factors, possibly combined with low image resolution, make barcode reading difficult in certain situations. Indeed, all existing image-based barcode readers have limited performance when it comes to images taken in difficult light conditions, or when the barcode is at a certain distance from the camera.

This paper presents a new algorithm for barcode reading that produces excellent results even for images that are blurred, noisy, and with low resolution. Quantitative comparisons on existing and new barcode image databases show that our technique outperforms other state-of-the-art softwares and compares favorably with other reported results.

A unique characteristic of our algorithm is that it never performs binarization of the graylevel brightness profile before processing. We argue that this early-commitment operation, executed by virtually all existing algorithms, translates into unrecoverable information loss, thus complicating all further processing. This is especially the case for low-resolution images, where binarization errors may have catastrophic effects. For example, Fig. 1 shows a challenging barcode, which would be hardly decoded using binarization.

This paper is organized as follows. After a review of the previous work in Sec. 2, we present our algorithm in Sec. 3. Comparative experimental tests are shown in Sec. 4, and the conclusions are given in Sec. 5. The Appendix describes a simple algorithm for barcode localization, which was necessary to extract the portion of the image to be decoded by our reader.

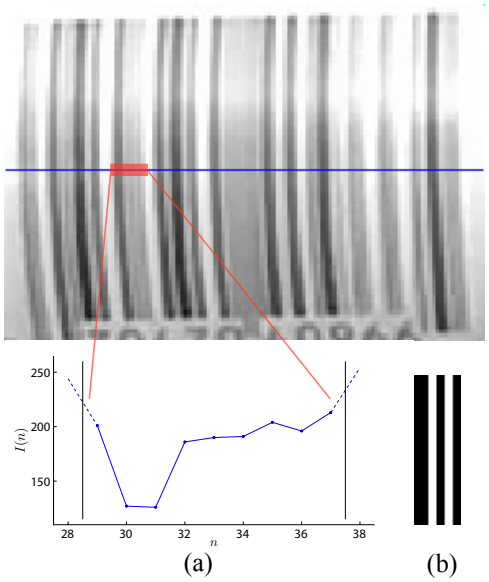


Figure 1. A challenging barcode image that is correctly decoded by our algorithm. The intensity profile from the segment highlighted in red on the blue scanline is plotted in (a). The underlying sequence of spaces and bars is shown in (b). Note how blur and low resolution affect the intensity profile. A system that binarizes the intensity would be hard-pressed to detect the correct pattern.

2. Previous work

Barcode decoding has been studied and optimized for decades and it now represents a consolidated industrial standard. (Pavlidis *et al*. provide an interesting analysis of this technology from an information theory standpoint [7].) Until recently, however, barcode reading was performed almost exclusively with dedicated, generally expensive hardware. Despite the rapid growth of interest in camera-based readers, most of the challenges posed by this new approach are yet to be solved.

Commercial scanners, such as those used in supermarkets, shine a light on the code and measure the intensity of its reflection, thus being virtually insensitive to ambient illumination. Mirrors then allow for the acquisition of multiple images of the code, which is also assumed to be fairly close to the scanner. As a consequence, the extracted scanlines are of very high quality.

Camera-based methods generally produce much lower quality scanlines. Moreover, the barcode generally does not completely fill the image; the first step to barcode interpretation is therefore its localization. We argue that this stage is intrinsically more robust to noise than the actual decoding, for it requires a lower accuracy. Existing methods for this step apply to the binarized image methods based on Hough transforms [4], edge orientation histograms [13], morphological operators [2], or wavelet transforms [12]. Other approaches simplify the problem assuming that the center of the image falls within the barcode area [5, 11].

Although decoding is a far more delicate process than localization, virtually all current techniques use some kind of binarization for this stage as well; even with the many successful thresholding algorithms that have been proposed [6, 8], a moderate amount of blur in the original image can dramatically affect the results (see Fig. 1).

Decoding can be performed by estimating the width of all the bars in the barcode from the binarized image [1, 2, 4]. Wachenfeld *et al*., instead, use an adaptive thresholding [11]. They then model the different digits and find the combination that best explains the scanline. Although they report a very high accuracy, they do not present any comparative test.

Krešić-Jurić *et al*. find potential edges differentiating the scanline and then use hidden Markov models to remove false positives [3]. Their method compares favorably with previous approaches although it was implemented on commercial barcode scanners. Tekin and Coughlan propose an elegant Bayesian framework for barcode decoding [9]. Their approach aims to find the edges of the bars in a fashion similar to Krešić-Jurić *et al*. [3] but they allow for both false positive and false negative. These methods are robust to a certain amount of noise in the data, however, they still need to make a hard decision as for where the edges are. Tekin and Coughlan made their database available and in Sec. 4 we compare our results with theirs.

Our main contribution is an algorithm capable of decoding 1-D barcodes from noisy pictures, whether the source of noise be motion blur or lack focus. Our method also proves effective on highly compressed pictures. Additionally, we propose

a simple method to segment the barcode out of the picture. Finally, we created a dataset of barcode images (which will be made public after publication) and evaluate our algorithm on it by means of comparison with publicly available barcode readers. The comparisons show that the proposed method outperforms existing methods, in particular as the quality of the image worsens.

2.1. UPC-A Barcodes - Syntax

UPC (Universal Product Code) is a technology to encode numbers with 12 decimal digits as an alternating sequence of black bars and white bars (*spaces*) with different widths. (The last digit is an error correcting check digit.) Each bar may have width equal to $r \times w$, where r (the *module width*) is an integer between 1 and 4, and w, the *base width* (sometime called *X-dimension*), is the width of the narrowest bar. The code is divided into two halves separated by a sequence of three spaces and two bars (*guard bars*), all of unitary module width. At the two ends of the barcode there is a sequence of two bars separated by a space, all of unitary module width (*start* and *end patterns*). The start and end patterns are separated from the edge of the barcode by a space of width equal to at least 9 times the base width, although this requirement is often violated in real-world instances. Between the start pattern and the guard bars, the code is divided into 6 equally spaced *digit segments*, each of which with length equal to 7 times the base width. Thus, the overall length of the barcode is equal to 95 base widths. Each digit segment represents one digit as a sequence of two spaces and two bars. The value k of a digit is encoded by the sequence of module widths $(r_1^k, r_2^k, r_3^k, r_4^k)$ of the bars and spaces in the digit segment. The standardized UPC-A sequences for the left half of the code are shown in Fig. 2. In the right half of the code, the same sequence of widths is used to encode a digit, however the role of spaces and bars is inverted.

3. The algorithm

Given an image containing a barcode, two distinct operations are needed for accessing the information contained in the barcode: *localization* and *reading*. Localization typically relies on the strong textural content of the barcode, without the need to exactly measure and interpret the width distribution of the bars. Reading can be performed on one or more scanlines, provided that they are within a certain angular span from the normal direction to the bars.

Although our work focuses on barcode reading, we implemented a simple and fast localization algorithm (described in the Appendix), that assumes that the bars are approximately vertical. This algorithm is by no means optimal but it works reasonably well in our studies, as it only serves as a necessary pre-processing stage to enable the experiments of Sec. 4. We also implemented other algorithms from the literature [13, 10] and none outperformed our simple method even with a higher computational load.

Our reading algorithm analyzes a single scanline contained in the detected barcode area. The only requirement is that the beginning and the end of the barcode pattern in the scanline are detected with a certain accuracy. For example, in our implementation we assume a localization tolerance in either end point equal to twice the width of the narrowest bar. Note that this task is much simpler than localizing all bars in the code, an operation that we avoid altogether. The UPC-A standard, in fact, requires that the initial and final bars be separated from the edge of the barcode by a *quiet area* of a width equal to at least 9 times the base width to facilitate localization.

Here is the algorithm outline. First, based on the detected endpoints of the scanline, we compute the spatial location of each digit segment in the barcode. For each of the 12 digits in the barcode, we compare the intensity profile of the corresponding segment of the scanline with binary templates, each representing a specific digit value as shown in Fig. 2. In order to account for inaccuracy in the localization of the spatial extent of each digit, we allow these templates to shift and scale in the horizontal direction. After defining a likelihood function to measure how well a deformed (shifted and scaled) template explains the observed intensity, one may be tempted to search for the deformation parameters that maximize it (that is, the shifted and scaled template that best explains the data), hoping not to end up in one of the many existing local maxima. We take a better, and theoretically more sound, route: we integrate the likelihood over the space of deformations, having defined a prior distribution of the deformation parameters. One important contribution of this work is to derive an algorithm that computes this marginalization integral *exactly* and in affordable computing time.

The scanline could finally be decoded choosing the digit values that maximize the likelihood of the data within each digit segment; however, this may lead to incorrect results due to noise, blur, or other causes. The risk of such errors can be reduced by exploiting global constraints on the overall sequence of digit values. The idea is that the "optimal" deformed templates should sit side by side on a scanline, without overlaps or gaps. We define a global cost function that, for each possible sequence of digit values, penalizes overlaps or gaps in the sequence of deformed templates, with the deformation parameters obtained by least squares regression. The minimum cost sequence can then be found via dynamic programming.

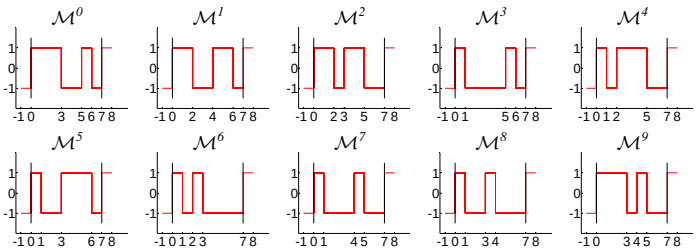


Figure 2. Each digit in a UPC-A code is encoded with a sequence of two bars and two spaces, represented in these graphs by values of 1 and -1.

Algorithmic details are provided in the next subsections.

3.1. Deformable models

We define a *model* (or *template*) $\mathcal{M}^k$ for digit k as a continuous piecewise constant function that alternates between -1 and 1, where a value of -1 (1) represents a black (white) bar (see Fig. 2). A model $\mathcal{M}^k$ for a digit in the left half of a UPC-A barcode begins with a '-1' segment and ends with a '1' segment, where both such segments have length of 1. The lengths of the i-th constant segment between these two end segments is equal to the module width r_i^k (as defined in Sec. 2.1.) A model is therefore an archetypical representation of one digit of a standardized scanline, which also includes one bar from each one of the nearby digits. These two additional bars have base width and known polarity; adding such bars to the template increases robustness of the matching process.

A parameterized model is a shifted and scaled (*deformed*) version of the original model:

$$\mathcal{M}^k_{o,w}(x) = \mathcal{M}^k((x-o)/w), \tag{1}$$

where o represents the starting point of the pattern and w represents the base width. (Note that models are functions of the continuous line, while the observation $I(n)$ is defined over the discrete space of pixels.) An example of deformed model is shown in Fig. 3.

3.2. Digit segment – conditional likelihood

Once the barcode has been localized in the image, and the endpoints (o_S, o_E) of the selected scanline have been estimated, the approximated position of each digit segment of the barcode is computed. More precisely, the j-th digit segment in the left side of the barcode is assumed to start at

$$o = o_S + 3w + 7w\ (j-1), \tag{2}$$

where:

$$w = \frac{o_E - o_S}{95} \tag{3}$$

and w is the estimated base width. These expressions derive from the fact that the overall length of the barcode is (ideally) equal to 95 times the base width, that each digit uses a segment (support) equal to 7 times the base width, and that the first 3 bars are guard bars.

We should stress the fact that, for a generic digit being considered, the value of o as computed in (2) is, in general, an incorrect estimate of the actual left edge of the digit segment, as a consequence of error in the estimation of the endpoints (o_S, o_E) and/or image distortion as due, for example, to perspective. However, suppose for a moment that the estimated location o and minimum bar width w are indeed correct. Then, in order to read the value of the digit, we could simply compare the intensity $I(n)$ within the segment with the models $\mathcal{M}^k_{o,w}$ for $0 \le k \le 9$, and pick the model that best fits the data. More precisely, we define the likelihood of the intensity within a generic digit segment when the digit takes a value of k (conditioned on o and w) as

$$p_k(I|o,w) \propto e^{-D(I,\mathcal{M}^k_{o,w})}, \tag{4}$$

where $I(n)$ represents the intensity profile of the considered scanline. The log-likelihood term D can be expressed as

$$D(I,\mathcal{M}^k_{o,w}) = \sum_{n=\lceil o-w \rceil}^{\lfloor o+8w \rfloor} D\left(I(n), \mathcal{M}^k_{o,w}(n)\right), \tag{5}$$

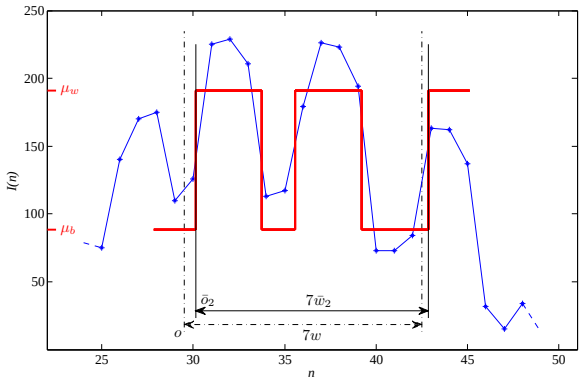


Figure 3. A sample of intensity profile in a scanline (blue line). The segment $[o, o + 7w]$ represents the location of the initial digit segment obtained from (2)-(3), whereas the segment $[\bar{o}_2, \bar{o}_2 + 7\bar{w}_2]$ is the estimated support segment as by (12) for $k = 2$. The red line represents the deformed model $\mathcal{M}^2_{\bar{o}_2,\bar{w}_2}$. For the sake of graphical clarity, the model was scaled in amplitude so that it alternates between μ_b and μ_w (as defined in Sec. 3.2).

where the variable n takes on only integer values (see Fig. 3). Note that this sum is computed over all pixels that fall within the segment $[o - w, o + 8w]$, which is the support of $\mathcal{M}^k_{o,w}(x)$.

A variety of functions can be considered for the log-likelihood D modeling the discrepancy between model and observation. We use the following robust formulation, which gave good results in the experiments. First, the quantities μ_w and μ_b representing the mean of the largest 50% and smallest 50% values of $I(n)$ are computed, along with the cumulative variance σ. Then,

$$D(I(n), -1) = \frac{[\max(I(n) - \mu_b, 0)]^2}{2\sigma^2} \tag{6}$$

and

$$D(I(n), 1) = \frac{[\min(I(n) - \mu_w, 0)]^2}{2\sigma^2}. \tag{7}$$

This function penalizes values of $I(n)$ that are small when $\mathcal{M}^k_{o,w}(n) = 1$ or large when $\mathcal{M}^k_{o,w}(n) = -1$.

3.3. Digit segment – total likelihood

A maximum likelihood approach based on the likelihood described above can be extremely sensitive to the even small errors of o and w. Such errors derive from the expected error in the estimation of the endpoints o_S and o_E. Assume for example that both o_S and o_E are computed with a tolerance of $\pm\Delta o$. Then, barring deformations or perspective effects, o has a tolerance of $\pm\Delta o$ as well, whereas w has a tolerance of $\pm 2\Delta o/95$. Thus, a method for taking uncertainty of the deformation parameters into account is necessary.

We approach this problem by first defining a probability density function $p(o, w)$ over the space of deformations. We then compute the total likelihood $p_k(I)$ by averaging $p_k(I|o, w)$ over such density:

$$p_k(I) = \int\int p_k(I|o, w)p(o, w)\, do\, dw. \tag{8}$$

Computing this integral may seem like a daunting task, especially if this needs to be performed in real time over an embedded computer such as a cell phone. On the contrary, we show that due to the particular nature of the model $\mathcal{M}^k$, and assuming a simple form for the prior $p(o, w)$, the integral in (8) can be computed *exactly* via numerical means with reasonably small complexity.

Our derivation exploits the fact that $D(I, \mathcal{M}^k_{o,w})$ is piecewise constant in the (o, w) space. This can be seen by breaking up the sum in (5) into six pieces, corresponding the segments in which $\mathcal{M}^k_{o,w}(x)$ takes on constant values of 1 or -1. More precisely, we notice that within segment $[d_i, d_{i+1}]$, where $d_i = o + w\sum_{l=0}^{i} r^k_l$ for $0 \le i \le 5$ and we set $r^k_0 = -1$ and $r^k_5 = 1$, the function $\mathcal{M}^k_{o,w}(x)$ is identically equal to $(-1)^i$.

$$D(I, \mathcal{M}^k_{o,w}) = \sum_{i=1}^{5} A_i, \tag{9}$$

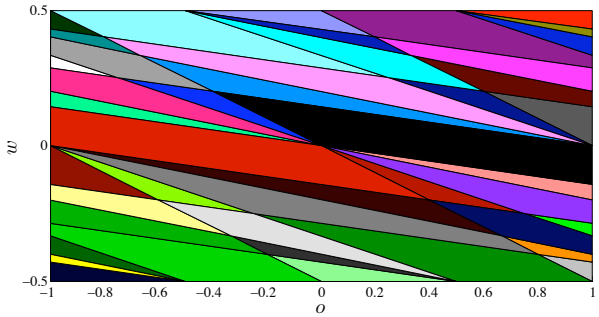


Figure 4. Partition of the space (o, w) into the cells $\{\mathcal{V}_2^t\}$ (*i.e.* for digit '2') within the rectangle $[-1, 1] \times [-0.5, 0.5]$. If (o_1, w_1) and (o_2, w_2) fall within the same cell, the conditional likelihoods $p_2(I|o_1, w_1)$ and $p_2(I|o_2, w_2)$ are identical. In other words, instead of computing the integral in (8) over every point of the space, the problem can be made tractable by only considering one point per cell without introducing any approximation (see Sec. 3.3).

with

$$A_i = \sum_{n=\lceil d_{i-1} \rceil}^{\lfloor d_i \rfloor} D(I(n), (-1)^i). \tag{10}$$

Hence, a variation of o or w determines a change of A_i (and therefore of $p_k(I|o, w)$) only when it causes d_{i-1} or d_i to cross over an integer value. Consequently, $p_k(I|o, w)$ is piecewise constant, and the integral in (8) becomes a sum. Next, we show how to compute the terms in this sum.

Let $\{\mathcal{V}_k^t\}$ be the minimum partition of the (o, w) plane such that $p_k(I|o, w)$ is constant within each cell $\mathcal{V}_k^t$ (with t representing the index of cells in the partition). Then

$$p_k(I) \propto \sum_t e^{-D_t} \int\int_{\mathcal{V}_k^t} p(o, w)\, do\, dw, \tag{11}$$

where $D_t = D(I, \mathcal{M}_{o,w}^k)$ for any (o, w) in $\mathcal{V}_k^t$. Note that the cells $\mathcal{V}_k^t$ are polygonal, as they are defined by the lines of equation $o + w(\sum_{l=0}^{i} r_l^k - 1) = s$, where s is any integer, and i is any integer between 0 and 5 (see Fig. 4). The list of cells $\{\mathcal{V}_k^t\}$, as well as the integral of $p(o, w)$ within each cell, can be computed offline and stored for online use. In fact, one easily sees that the cells form a periodic pattern (with period equal to 1 both in o and w), hence only the cells within such a period need to be stored.

Regarding the implementation of this procedure, the following observations are in order:

1. The computation of the likelihood in (11) can be sped up by precomputing the sequences $D(I(n), 1)$ and $D(I(n), -1)$. Then, for each cell, one only needs to add together selected samples from the two sequences.

2. At run time, a specific set of cells is chosen from the list based on the tolerance Δo and Δw on the estimated values of o and w, which are easily derived from the tolerance of the estimated endpoints o_S and o_E. More precisely, we compute the sum in (11) over the cells that intersect the rectangle with sides $[o - \Delta o, o + \Delta o]$ and $[w - \Delta w, w + \Delta w]$, where o and w are estimated as by (2).

3. The integration of $p(o, w)$ within each cell results particularly simple if $p(o, w)$ is assumed to be uniform within the considered rectangle in the (o, w) space. In this case, the integral is proportional to the area of the polygonal cell, which can be easily computed and stored offline. In our implementation we made use of this simple, yet effective model.

As will be shown shortly, it is also useful to estimate, for each possible digit value k, the deformation parameters (o, w) given the intensity profile $I(n)$ within a digit segment. We choose the least squares estimator $(\bar{o}_k, \bar{w}_k)$ of these quantities (under the density $p(o, w)$), which is given by the conditional expectation. Using Bayes rule, this is equivalent to

$$\begin{aligned}(\bar{o}_k, \bar{w}_k) &= \int\int (o, w)\, \frac{p_k(I|o, w) p(o, w)}{p_k(I)}\, do\, dw \\ &\propto \frac{1}{p_k(I)} \sum_t e^{-D_t} \int\int_{\mathcal{V}_k^t} o\, w\, p(o, w)\, do\, dw.\end{aligned} \tag{12}$$

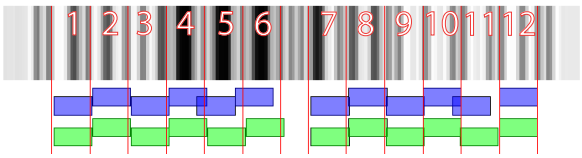


Figure 5. The support segments $[\bar{o}_{j,k(j)}, \bar{o}_{j,k(j)} + 7\bar{w}_{j,k(j)}]$, where $k(j)$ are the maximizers of the total likelihood $p_{j,k(j)}(I)$ for each digit index j, are shown in blue against the support segments corresponding to the sequence of values $\{k\}$ minimizing the global cost C in (14), shown in green. Digits 5, 6, and 11 are not correctly decoded and their position is therefore miscalculated (blue). The algorithm described in Sec. 3.4 successfully enforces global consistency and, thus, correct decoding (green). The red lines represent the original digit segments, obtained from (2)-(3). In order to provide a visual intuition of the intensity profile of the scanline under consideration, the latter was repeated vertically to form a graylevel image.

The integrals in the above equation can be precomputed and stored for online use. If the assumption of uniformly distributed (o, w) is made (as in point 3. above), then the terms in the sum are the centroids of the cells $\{\mathcal{V}_k^t\}$.

3.4. Imposing spatial coherence

Our model makes the simplifying initial assumption that the digit segments are equally spaced (see (2)-(3)). This also implies that the base width w is constant across the barcode. In practice, we should expect that the digit segment length may vary from segment to segment, hopefully within the confidence intervals Δo and Δw. Ideally, however, the segment representing a given digit in the scanline (as computed from the estimates $\bar{o}_k$ and $\bar{w}_k$) should be adjacent to (but non overlapping with) the neighboring segments. The choice of an incorrect value of k due to single-digit analysis is likely to result in a supported segment that does not fit well together with the other segments (see Fig. 5). This observation can be exploited by imposing a global constraint as follows.

Suppose that the j-th digit takes value $k(j)$. (Note that we need to make the dependency on j explicit in our notation from now on.) The estimated deformation parameters $(\bar{o}_{j,k(j)}, \bar{w}_{j,k(j)})$ define the supported segment $[\bar{o}_{j,k(j)}, \bar{o}_{j,k(j)} + 7\bar{w}_{j,k(j)}]$. We define the overlap/gap extent between the j-th and $(j+1)$-th estimated digit segments as

$$O_{j,k(j),k_{j+1}} = |\bar{o}_{j,k(j)} + 7\bar{w}_{j,k(j)} - \bar{o}_{j+1,k(j+1)}|. \tag{13}$$

Now define a global cost function as follows:

$$C(\{k\}) = \sum_j \alpha O^2_{j,k(j),k(j+1)} - \log p_{j,k(j)}, \tag{14}$$

where α is a balancing parameter, and the sum extends to all digits in the left and right half of the barcode. (α was set to be equal to 0.1 in our experiments). The cost function in (14) penalizes sequences of digit values that create large overlaps or gaps between two consecutive digit segments or that produce low values of likelihood. Dynamic programming can be used to minimize the cost function C over the space of sequences $\{k\}$ of digit values. Fig. 5 shows the outcome of the application of this technique.

4. Implementation and tests

We implemented and tested our algorithm in Matlab. The simple localization algorithm described in the Appendix was used to provide a scanline segment input to our reader. The maximum tolerance Δo at the endpoints o_S and o_E was set to $\pm 2w$, where w, the initial estimate of the base width, was defined in (3). In practice, this means that we expect the localization algorithm to possibly miss one the first bar in the start and end pattern. The minimum scanline width $o_E - o_S$ that our algorithm was able to decode was of 100 pixels. Note that this corresponds to a base width of only 1.05 pixels.

The computational speed of the algorithm depends heavily on the scanline width. This is mostly due to the fact that the number of cells V_k^t depends on the tolerances Δo and Δw, which are proportional to the scanline width. For example, when the scanline width is equal to 100 pixels, then 25 cells are generated. However, in the case of a high resolution image of a barcode at short distance, producing a scanline width of 1128 pixels, 2185 cells are generated. In order to reduce the number of cells to consider, we implemented a simple variation of the algorithm, by fixing the width of the first and last bars in the model to the minimum width considered. Remember that these are "overlap" bars with nearby digits, and that they always have unitary width in the model $\mathcal{M}^k$. With this modification, the number of cells is reduced to 17 in the 100 pixel

Data set "Clean" from [9]				
Our algorithm	[9]	DataSymbol	DTK	Total
43	42	39	**43**	44

Data set "Hard" from [9]				
Our algorithm	[9]	DataSymbol	DTK	Total
19	2	0	1	35

Data set 1			
Our algorithm	DataSymbol	DTK	Total
43	26	42	62

Data set 2			
Our algorithm	DataSymbol	DTK	Total
10	4	6	10

Data set 3			
Our algorithm	DataSymbol	DTK	Total
9	0	0	20

Figure 6. Comparative results showing the number of barcodes correctly detected in the different data sets considered. The last column reports the total number of images in each data set.

scanline width case, and to 641 in the 1128 pixel case. The computational speed of the overall reader (excluding the original segmentation) ranges between 0.076 seconds to 0.65 seconds.

In order to assess the performance of the system, we tested it on a variety of images. Unfortunately, we could find only one barcode image database for comparative assessment[1]. This database, which was created by Tekin and Coughlan, is accessible at `www.ski.org/Rehab/Coughlan_lab/Barcode`. The images are all of high resolution, and each image was manually cropped around the barcode. The authors divided it into "Hard" and "Clean" subsets, and showed results of their algorithm on both sets [9].

In order to assess our system in other realistic situations, we gathered a number of images taken from two different cellphones, and created three new data sets. Data set 1 contains images at high resolution (1024×768) from a Nokia N95 cell phone. This device has autofocus capability, although not all images collected were properly in-focus, and some had some amount of motion blur. Data set 2 contains images taken by the same cell phone, but at 640×480 resolution, and highly compressed in JPEG (with apparent coding artifact). Data set 3 contains images at 1152×864 resolution taken with an older Nokia 7610 phone, with fixed focus. All three data sets have multiple pictures of several barcodes, taken from different distances and in different conditions.

Our algorithm was tested against the algorithm of [9], for the "Hard" and "Clean" data set, as well as against two barcode reading softwares that are available online. The first one, from DataSymbol[2], was also considerd in [9]. The second one, from DTK[3], was shown to produce impressive results. A third online software, from QualitySoft, was considered in [9], but we neglected comparison with it since it gives very poor results.

Numerical results from our tests are presented in Fig. 6. It is shown that in all the data sets, our algorithm outperforms the other techniques. In particular, our algorithm performs comparatively well for the most challenging sets (Data set "Hard" and Data set 3). A sample of images correctly decoded by our algorithm is shown in Fig. 7, while examples of failures are shown in Fig. 8. Note that in many cases, failure was due to incorrect initial localization due to poor segmentation. The reader is invited to zoom-in to notice how poor the image quality is for many of the pictures that our algorithm correctly decodes.

5. Conclusions

We have presented a new algorithm for barcode reading that can deal with images that are blurred, noisy, and with low resolution. Unlike previous approaches, this algorithm does not binarize the image, thus sidestepping a critical and often error-prone early-commitment procedure. We use deformable templates for representing each digit of the barcode, integrating over the set of all expected deformations. A final procedure for global spatial coherence helps reducing the risk of errors at the individual digit level. The experimental results show improved performance with respect to other state-of-the-art

[1] The authors of [11] claim to have assembled a barcode image database for public use, but we have not been able to access it.

[2] `http://www.datasymbol.com`

[3] `http://www.dtksoft.com`

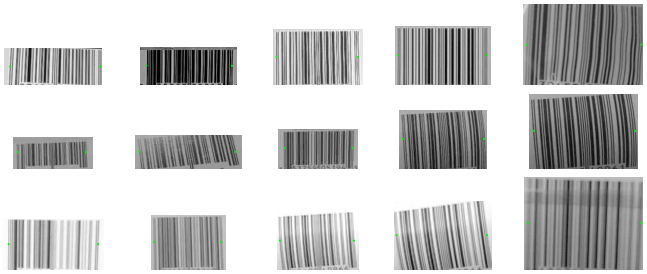


Figure 7. Example of barcodes correctly decoded by our algorithm from our three data sets. Each image shows the segment extracted by the localization algorithm described in the Appendix. The green stars indicate (o_S, o_E), the original estimates for the endpoints of the scanline.

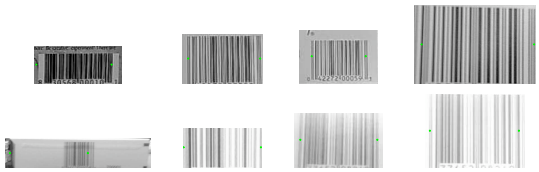


Figure 8. Example of images in which our algorithm fails from Data Set 1 and 3. The green stars indicate (o_S, o_E), the original estimates for the endpoints of the scanline.

software and algorithms, especially for the most challenging images. Our Matlab implementation of the overall system runs reasonably fast; porting of the algorithm to a Symbian cellphone is in progress.

The most obvious improvement to the algorithm would be to consider non-horizontal lines (for rotated barcodes) and multiple scanline analysis. The spatial coherence constraint could be easily extended to the multiple scanlines case. Another issue worth further consideration is the model for the greylevel values. Our current model is bimodal—each pixel is assumed to be black or white. In fact, the model could be improved to account for the point spread function of the optical system, which smears the edge of a bar across a few pixels. This enhanced model would be particularly useful for low resolution imagery, where even a small amount of blur may completely wash out the narrower bars.

References

[1] R. Adelmann, M. Langheinrich, and C. Flörkemeier. A Toolkit for Bar-Code Recognition and Resolving on Camera Phones–Jump Starting the Internet of Things. In *Workshop Mobile and Embedded Interactive Systems (MEIS06) at Informatik*, 2006.

[2] D. Chai and F. Hock. Locating and decoding EAN-13 barcodes from images captured by digital cameras. pages 1595–9, 2005.

[3] S. Krešić-Jurić, D. Madej, and F. Santosa. Applications of hidden Markov models in bar code decoding. *Pattern recognition letters*, 27(14):1665–1672, 2006.

[4] R. Muniz, L. Junco, and A. Otero. A robust software barcode reader using the hough transform. *Information Intelligence and Systems, 1999. Proceedings. 1999 International Conference on*, pages 313–319, 1999.

[5] E. Ohbuchi, H. Hanaizumi, and L. Hock. Barcode readers using the camera device in mobile phones. In *Proceedings of the Third International Conference on Cyberworlds (CW'04)*, volume 00, pages 260–265, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[6] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–6, 1979.

[7] T. Pavlidis, J. Swartz, and Y. Wang. Fundamentals of bar code information theory. *Computer*, 23(4):74–86, 1990.

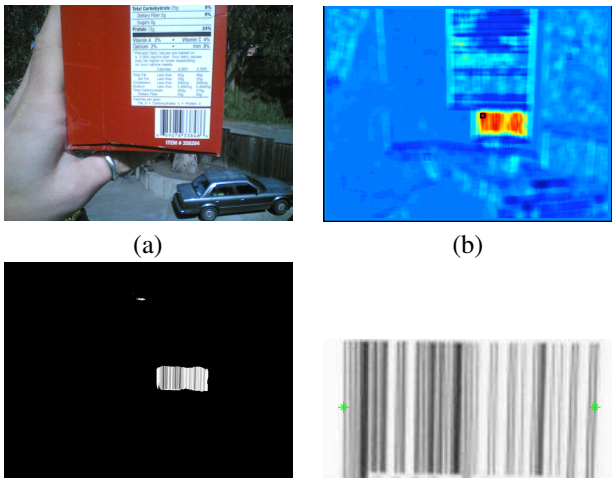
[8] Y. Solihin and C. Leedham. Integral ratio: a new class of global thresholding techniques for handwriting images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(8):761–768, Aug 1999.

[9] E. Tekin and J. Coughlan. A bayesian algorithm for reading 1d barcodes. In *To appear in Sixth Canadian Conference on Computer and Robot Vision*, 2009.

[10] A. Tropf and D. Chai. Locating 1-D Bar Codes in DCT-Domain. In *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings*, volume 2, 2006.

[11] S. Wachenfeld, S. Terlunen, and X. Jiang. Robust recognition of 1-d barcodes using camera phones. In *International Conference of Pattern Recognition*, pages 1–4, 2008.

[12] K. Wang, Y. Zou, and H. Wang. 1d bar code reading on camera phones. *International Journal of Image and Graphics*, 7(3):529–550, July 2007.



(a) (b)

(c) (d)

Figure 9. An example of barcode localization with our algorithm. (a): Original image. (b): The smoothed map $I_s(n)$. (c): Binarization by thresholding of $I_s(n)$. (d): The resulting rectangular segment, along with the selected scanline and the endpoints o_S and o_E marked by stars.

[13] C. Zhang, J. Wang, S. Han, M. Yi, and Z. Zhang. Automatic real-time barcode localization in complex scenes. In *International Conference of Image Processing*, pages 497–500, 2006.

Appendix

In this Appendix we briefly describe the algorithm used in our experiments for the localization of barcodes with approximately vertical bars. This simple technique is meant solely as a tool to enable us to test the barcode reader, which is the main contribution of this work. Hence, we do not make any claims regarding its performance, except that it served reasonably well for our purposes.

The idea behind the algorithm is that, in correspondence of a barcode (with approximately vertical bars), one should expect an extended region characterized by strong horizontal gradients and weak vertical gradients. Accordingly, we first compute the horizontal and vertical derivatives, $I_x(n)$ and $I_y(n)$, at each pixel. Then we combine them together in a non-linear fashion as by

$$I_e(n) = |I_x(n)| - |I_y(n)| \tag{15}$$

It is reasonable to assume that many points within a barcode should have a large value of $I_e(n)$. We run a block filter of size 31×31 over the field $I_e(n)$, obtaining the smoothed map $I_s(n)$. Note that block filtering can be computed with only few operations per pixel. Finally, we binarize $I_s(n)$ with a single threshold, selected using method proposed by Otsu [6]. As a consequence of thresholding, the map $I_s(n)$ may contain more than one blob. Rather than computing the connected components of the thresholded map, we simply select the pixel n_0 that maximizes $I_s(n)$, under the assumption that the correct blob (centered at the barcode) contains such pixel. This hypothesis was always verified in our experiments. Then, we expand a vertical and an horizontal line from n_0, and form a rectangle (with sides parallel to the axes) containing the intersections of these lines with the edge of the blob. The horizontal line through the center of this rectangle is chosen as the scanline for the analysis. In order to localize the endpoints o_S and o_E of the barcode, we first determine the intersections of this scanline with the rectangle computed earlier. Then starting from each end, we proceed inwards until pixel with significant negative value of the derivative is found, which is taken to represent the left (right) edge of the first (last) bar of the start (end) pattern.

This algorithm was implemented in Matlab. For a 640×480 image, the execution time was of approximately 0.065 seconds, while for a 1152×864 it was of approximately 0.21 seconds.